**SCAT (suspicious commit analysis tool)**

Code review is a critical part of application security - it requires malicious behaviour or insertion of backdoors to require conspiracy (or drastic oversight) in addition to a single malicious actor.

Building a machine-enforced code review system is a necessary task for any large tech company. Much software already exists to mediate this task, such as Gerrit.

Organizations tend to (need to!) trust their engineers, but as a failsafe, it could be helpful to have a smart gerrit plugin that analyzes commits and reviews to check if anything is out of place. Did code get reviewed by someone who doesn’t tend to work in or review the relevant files? Was a review carried out by someone who doesn’t tend to review the other person’s code.

This tool analyzes a git log to develop a model as to which developers tend to work in specific files in a code base, and who tends to review each other’s code. While this could be used to suggest possible reviewers for a change, a much more valuable service is to flag unusual commits or reviews for double check by a ‘trusted reviewer’ before that commit is pushed.

Notably, this is not only useful against insider threats, but also against account hijacking attacks, where it may be much easier for attackers to compromise two accounts than to compromise two specific/adjacent accounts (or even know which developers would be appropriate to compromise).

**Components:**

parser.py - parses log files or commits to generate ‘commit’ objects.

model.py - takes in a list of commit objects, generates model, using scikit and pandas

analyzer.py - contains functions to take in a single commit, parse it, and assign it a (to be renamed) ‘sketchiness’ rating.

**Commit Objects:**

* Timestamp
* Author
* Files
* Reviewer
* Review Timestamp